Recitation Session Solutions, October 25 2017

Ex2.

To prove:

We want to prove `P(list)` for any `list` of type `List[Int]`, where `P(list)` is defined as:

`P(list) := list.foldLeft(z)(add) === z + sum(list),` for all `z` of type `Int`

The proof proceeds by structural induction on `list`.

Case `Nil`:

We want to show `P(Nil)`.
Let `z` be an arbitrary expression of type `Int`.

```
Nil.foldLeft(z)(add)    ===   (by 3)    z
                        ===   (by 8)    z + 0
                        ===   (by 1)    z + sum(Nil)
```

Which proves `P(Nil)`.

Case `x :: xs`:

We want to show `P(x :: xs)`, assuming `P(xs)`.

Induction hypothesis: `P(xs)`

`(IH)   xs.foldLeft(z’)(add) === z’ + sum(xs),` for all `z’` of type `Int`

Let `z` be an arbitrary expression of type `Int`.

```
(x :: xs).foldLeft(z)(add)
                ===   (by 4)     xs.foldLeft(add(z, x))(add)
                ===   (by IH)    add(z, x) + sum(xs)
                ===   (by 5)     (z + x) + sum(xs)
                ===   (by 7)     z + (x + sum(xs))
                ===   (by 2)     z + sum(x :: xs)
```

Which proves `P(x :: xs)`.

This completes the proof.